Real-Time Data: Frequently Asked Questions

Tim Getsch
Paul Cornell
Microsoft Corporation

July 2001

Applies to:
Microsoft® Excel 2002

Summary: This article provides answers to frequently asked questions about the real-time data feature of Microsoft Excel 2002. (13 printed pages)

Contents

What is Real-Time Data (RTD)?

Real-time data, often referred to as RTD, is data that updates on its own schedule (for example, stock quotes, manufacturing statistics, Web server loads, and warehouse activity). It is something that people have tried to construct with previous versions of Microsoft Excel, but there have been many difficulties and limitations. Microsoft Excel 2002 is now designed to handle such data.

What Are Some Target Scenarios for RTD?

Live Data Scenarios

Examples of Live Data Scenarios include stock market data, manufacturing statistics, Web server loads, and warehouse activity.

A Specific Scenario (Equities-Trading Firm)

An equities-trading firm wants to do calculations in Excel based on real-time stock market data. They have experimented with Dynamic Data Exchange (DDE), semi-volatile functions, and other random methods of getting real-time data in Excel 97 and 2000. To date, they haven't been happy with any of them for various reasons. Now that they have Excel 2002, they're able to get real-time data into Excel reliably and with the throughput necessary to handle the calculation models they use.

Asynchronous Data Scenarios

Examples of Asynchronous Data Scenarios include database queries, complex calculations, and slow data retrieval.

A Specific Scenario (MSN MoneyCentral Stock Quotes Add-In)

People have been able to use Web Queries since Excel 97 to get refreshable stock market data in Excel. But there are a lot of limitations with Web Queries, and often what people really want is just to be able to put a specific piece of information in a specific cell and have that be refreshable. They don't want to be limited by the structure and amount of the data that is returned by the Web Query. Since the connection speed and network traffic can often slow down access to Web pages, it is nice that Web Queries can be done asynchronously so that Excel doesn't have to be locked. By building an RTD server, it was possible to develop an add-in that offered a function that could be inserted in a specific cell and, like Web Queries, be refreshed asynchronously.

What Are the Design Goals for RTD?

RTD offers the following advantages:

- Has Excel function-like syntax.
- Puts the real-time data in a cell.
- Uses cell references as part of the formula.
- Updates in real time.
- Is extremely efficient.
- Does not drop updates; modal dialog boxes shouldn't have any effect.
- Is able to drive calculation; formulas based on cells that reference real-time data should update accordingly.
- Keeps track of the state and location of real-time data formulas so that the server doesn't have to.

Does Microsoft Office XP Come with Any RTD Servers? Is Microsoft Building Any RTD Servers?

Not right out of the box, but there will be an add-in called MSN® MoneyCentral™ Stock Quotes on the Office Update Download Center that uses the new RTD architecture to make it possible to asynchronously get stock quotes from

the MoneyCentral Web site. (To download the Stock Quotes add-in, go to the Office Update Download Center. To learn how to use the Stock Quotes add-in, go to the Office Assistance Center). Note that the Stock Quotes add-in only uses a fraction of the RTD features, and it isn't providing real-time data in the traditional sense. Instead, the add-in offers a button that tells it to fire off a request to MoneyCentral. When the request returns, the RTD server that is built into the add-in lets Excel know that it has new data. When Excel is ready for the new data, it calls into an interface on the RTD server, and the RTD server hands over the new values.

Programmatically, What is an RTD Server? Is it a Dedicated Computer That Serves Real-Time Data?

Basically an RTD server can be a dynamic-link library (DLL), an executable program (EXE) that is executed on your computer, or an EXE that runs on a separate computer. Technically, it is an Automation server that implements an extra interface called **IRtdServer** that communicates with Excel. So, a dedicated computer is not required, but a computer may be set aside just for the purpose of running instances of one or more RTD servers through DCOM.

What Do I Need to Build an RTD Server?

The needs for different types of systems vary, but typically there is a feed of data (for example, manufacturing statistics, stock quotes, Web server loads, and so on). By using Microsoft Visual Basic® or Microsoft Visual C++®, all you need to do is build the RTD server which conceptually translates the language of the feed to a language that Excel now understands. If you are building an RTD server to run on each client's computer, you can use any edition of Visual Basic or Visual C++ that can build Automation servers. If you want to build an RTD server that runs on a separate computer through DCOM, then you need the Enterprise Edition of Visual Basic or Visual C++.

A lot of the people that we expect to build RTD servers already have their own RTD solutions and all of these solutions have major limitations. The new RTD architecture should be a relatively simple way to interface between their existing feed and Excel without any of those limitations.

What Are Some of the Limitations of Existing Real-Time Solutions?

Some of the common limitations of real-time solutions include the following: updates get missed easily, updates don't always trigger a calculation, solutions use non-standard Excel function syntax, all functions must be hard coded, and the solutions are inefficient.

Existing real-time solutions rely on either a push mechanism or a pull mechanism, and both of these have big limitations.

What is a Push Mechanism?

With a push mechanism, the part of the solution that is feeding the real-time data tries to push the data into Excel. One of the disadvantages of push mechanisms is that they often try to push data into Excel when Excel is not ready for it (for example, while Excel is doing a calculation or has a modal dialog box that needs to be handled). This often leads to dropped updates and sometimes even causes a crash.

What is a Pull Mechanism?

With a pull mechanism, the part of the solution that is in Excel (for example, an add-in) pulls data from the feed. One of the disadvantages of pull mechanisms is that they repeatedly make requests to the feed even if there haven't been any updates.

Why is the New RTD Architecture in Excel So Much Better?

The new RTD architecture is a fully functional, first-class feature built into Excel 2002. The architecture eliminates the need for workarounds or hacks to get data into Excel asynchronously. The architecture is fast, efficient, reliable, and robust. And to top it all off, RTD servers are fairly easy to build.

What Mechanism Does the New RTD Architecture Use?

RTD uses a hybrid push and pull mechanism. This effectively gives it all of the advantages of both push and pull mechanisms and none of the disadvantages of either mechanism.

You can think of this hybrid push and pull mechanism as a polite conversation between the RTD server and Excel. When the RTD server wants to give Excel an update, it notifies Excel by giving it a tap on the shoulder (this is the push). The RTD server can tap Excel on the shoulder whenever and as frequently as it wants. Excel just notes that the RTD server wants to give it an update. When Excel is ready for the update, it grabs the update from the RTD server (this is the pull).

What is Wrong With Dynamic Data Exchange (DDE) for Real-Time Data?

DDE has often been used to try to implement real-time data in Excel, but it has several limitations. Here are some of them:

- DDE has a different function format from standard Excel functions.
- DDE can't use cell references in a function; everything must be hard coded.
- DDE gets confused when things are happening in Excel (for example, a calculation is taking place, a dialog box needs to be handled, and so on).
- DDE wasn't designed for getting asynchronous data in Excel in a robust and high-performance way.

Is RTD Functionality Available in the Office XP Web Components?

No, the RTD functionality is not available in the Office XP Web Components.

How Does the New RTD Architecture Facilitate Updates?

When the RTD server wants to update any values, it calls the **UpdateNotify** method on a new interface that Excel has implemented as **IRTDUpdateEvent**. This can be called any time that the RTD server wants and as frequently as it wants. This is like tapping Excel on the shoulder. If Excel is busy, it just notes that it has been notified and get back to the RTD server when it is not so busy. If the RTD server calls the **UpdateNotify** method numerous times while Excel is busy, that is fine. Excel knows that when it gets a chance, it has to get new values from that RTD server. When Excel is ready, it calls the **RefreshData** method on the **IRtdServer** interface that the RTD server implements. This is where the RTD server gives Excel as many updated values as it wants to. Excel then updates all of the corresponding **=RTD()** formulas.

What Kind of Performance Statistics Have Been Seen With the New RTD Architecture?

On a Pentium III 500 MHz processor with 128 MB of RAM, 20,000 unique topics can be updated three times per second; one topic can be updated 200 times per second.

How Frequently Can Excel Accept Updates?

These are the numbers that we get with an RTD server running on the same computer that Excel is on.

Based on a Pentium III 500 MHz processor with 128 MB of RAM, 20,000 unique topics can be updated three times per second; one topic can be updated 200 times per second.

The number of times that a single topic can be updated seems to be limited by the number of times that Excel checks for Microsoft Windows® messages, which is at most 700 times per second. Since some of the messages have higher priority than RTD does, Excel effectively gets about 200 updates per second.

How Frequently Can the RTD Server Give Updates?

If the RTD server is running on the local computer, the limitation is a combination of the efficiency of the RTD server and the number of times that Excel requests updates. For more information, see the section How Do I Configure the RTD Throttle Interval in Excel? below.

If the RTD server is on a different computer, the throughput is limited by DCOM. Each instance of Excel using that RTD server instantiates a COM object on the computer that has the RTD server. So, the total RTD throughput of a computer that is designated to run these RTD servers is limited by the number of RTD servers (COM objects) that can be instantiated on that computer, as well as the network traffic.

What Happens When the Spreadsheet Cannot Process Incoming Requests Fast Enough (For Example, Either the Load is Too Heavy Or Excel is Too Slow)? Will the Data Get Dropped?

This is not an issue because the RTD feature uses a hybrid push and pull mechanism. This would be a problem if Excel relied purely on a push mechanism. The way RTD works is that the RTD server can tell Excel that it has updates as frequently as it wants (the push portion), but Excel doesn't pull the new values from the RTD server until it is ready to handle them. Hence, the above-mentioned state will never occur.

What Kind of Connection Exists Between the RTD Server and the Excel Spreadsheet?

The RTD server is just a COM object that implements the **IRtdServer** interface. Excel has added an **IRTDUpdateEvent** interface. The RTD server and Excel communicate with each other by calling into these interfaces. This is done through COM when the RTD server is running on the local computer, and it is done through DCOM when the RTD server is on a separate computer.

Is it Possible to Connect to an RTD Server Outside of the Firewall?

Best performance will likely be obtained if the RTD server is executed on the local computer or anywhere inside of the firewall, and is connected to a feed that is outside of the firewall. For example, a company could develop an RTD server for their customers that is installed on each customer's computer, and that RTD server could communicate with the company's Web site to get data.

However, Marc Levy's article COM Internet Services discusses how the Tunneling Transmission Control Protocol (TCP) "allows a client and a server to communicate in the presence of most proxy servers and firewalls, thereby enabling a new class of COM-based Internet scenarios."

Why Isn't XML Returned By the RefreshData Method Instead of This Array?

Efficiency. Since the RTD server and Excel are communicating directly with one another, there is no need for the overhead of constructing and parsing the XML. It is much more efficient to just pass the array of topic IDs and values to Excel.

When Does Excel Check for Updates?

Excel never really checks for updates—it only gets updates after the RTD server has told Excel that it has updates. Excel only gets updates when it is in a "good state" and it waits at least the number of milliseconds specified by the RTD throttle interval. Excel does not get updates while a modal dialog box is displayed, while a cell is being edited, or while it is busy doing other things. Basically, when Excel is not totally swamped and it is in a state where cell values can change, it glances at its "RTD clock." If the throttle interval has passed, it calls the **RefreshData** method on each of the RTD servers that has notified it of an update by calling its **UpdateNotify** method.

How Do I Configure the RTD Throttle Interval In Excel?

This can only be modified via the Excel object model or the registry. There is no user interface for configuring the RTD throttle interval in Excel.

- If the RTD throttle interval is set to -1, this is considered manual mode, and Excel only checks for updates when `Excel.Application.RTD.RefreshData` is called.
- If the RTD throttle interval is set to zero, Excel checks for updates every chance it gets.
- If the RTD throttle interval is set to something greater than zero, Excel waits at least that number of milliseconds between checks for updates.

> **Caution** If updates come in so frequently that Excel is continuously updating values and doing calculations, Excel might end up in a state where it never gives the user a chance to do anything, effectively getting in a hung state. If this happens, set the Excel throttle interval higher.

To set the throttle interval higher through the Excel object model:

1. In Excel, go to the Visual Basic Editor (by pressing ALT+F11 or clicking **Visual Basic Editor** from the **Macro** menu (**Tools** menu)).
2. In the **Immediate** window (press CTRL+G or click **Immediate Window** on the **View** menu), type this code:
   `Application.RTD.ThrottleInterval = 1000`
3. Make sure your cursor is on the line that you just typed, and then press ENTER.
4. To verify that it is set correctly, type this line of if code in the **Immediate** window:
   `? Application.RTD.ThrottleInterval`
5. If you put your cursor at the end of this line and press ENTER, it should display `1000`. Then you know that your throttle interval is set correctly.

To set the throttle interval higher through the registry, set the following registry key. It is a DWORD and is in milliseconds:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Excel\Options\RTDThrottleInterval
```

> **Note** The throttle interval works in conjunction with an **IRTDUpdateEvent** callback object. If the **IRTDUpdateEvent** callback object is never called, then it doesn't matter what your throttle interval is set to; Excel never asks for the updated data.

How Do I Build An RTD Server That Runs On a Separate Computer From the One Excel is Running On?

See the Knowledge Base article Use an Excel RTD Server with DCOM.

How Do I Hide Some of the Parameters to the RTD Function?

It is possible to write a function wrapper that hides the programmatic ID (ProgID), hides the server, and maybe masks the topics. If such a function wrapper is created, that function can be used instead of the RTD function and you dont have to remember all of that stuff. For more details, see the next question.

How Do I Create a Function Wrapper?

A function wrapper is a user-defined function (UDF) that calls another function. UDFs can be created in Visual Basic for Applications (VBA) or in an Automation add-in.

If the **RTD** function looks like this:

```
RTD("MyRTDServer.ProgID", "MyServer", "StockQuote", "MSFT", "NASDAQ", "LAST PRICE")
```

Then the following function wrapper could hide some of these details:

```
Function StockQuote(Symbol, Market, Property)
    StockQuote =    Excel.Application.WorksheetFunction.RTD("MyRTDServer.ProgID", "MyServer", _
    StockQuote, UCase(Symbol), UCase(Market), UCase(Property))
End Function
```

With this UDF, users only have to enter the following function:

```
StockQuote("MSFT", "Nasdaq", "last price")
```

They don't have to worry about consistent capitalization, and they dont have to remember any of the extra parameters to the **RTD** function.

What is the GetNewValues Parameter of the ConnectData Method For?

In short, set the *GetNewValues* parameter to **True** if you always want the initial value of your **RTD** function to be

the return value of the **ConnectData** method, and leave the *GetNewValues* parameter alone if you want Excel to use the previous value that it had saved with the Excel spreadsheet.

When a workbook with **RTD** functions is saved as an Excel spreadsheet, then the values of these functions are saved with the workbook. When such a workbook is opened, the **ConnectData** method is called for each new **RTD** function. If the *GetNewValues* parameter is set to **True**, the saved value is discarded and the return value from the **ConnectData** method is used. If the **GetNewValues** parameter is set to **False**, the saved value is used until that topic has been updated and the return value from the **ConnectData** method is ignored. By default, the *GetNewValues* parameter is **False** if Excel has a saved value, and it is **True** if Excel doesn't have a saved value.

> **Note** It is not possible to find out the saved value of the **RTD** function from within the **ConnectData** method.

How Do I construct the Return Array For the RefreshData Method?

The return array for the **RefreshData** method must be a two-dimensional **Variant** array where the first dimension goes from zero (the *TopicID* parameter) to one, and the second dimension goes from zero to TopicCount minus one. Here is an example:

```
Dim ra()

TopicCount = UpdatedTopics.Count
ReDim ra(0 To 1, 0 To TopicCount - 1)
i = 0
For Each Item in UpdatedTopics
    ra(0, i) = Item.TopicID
    ra(1, i) = Item.Value
    i = i + 1
Next Item
```

How Do You Manage a Group of Topics From the RTD Server?

Sometimes topics need to be treated as a group. For example, the Last Price, Size of Last Sale, and Time of Last Sale for a stock should all correspond to on another. You don't want Last Price and Size of Last Sale to be from different trades.

Here are three key things to ensure that a group of topics stays in sync:

1. Always set the *GetNewValues* parameter to **True** in the **ConnectData** method. Otherwise, saved values may get mixed with new values.
2. Only return a value from the **ConnectData** method if the value corresponds to the values that have already been returned for the topics in that group.
3. In the **RefreshData** method, all topics in the group should be updated at once. Unless a value hasnt changed, each topic in the group should be included in the **RefreshData** method's return array.

How Do I Make Sure That Excel sees Every Update?

It's basically up to the RTD server to decide what to do with the values if more than one update on a particular topic comes in before Excel calls back to the server for updates. Generally, we expect servers to just throw out the old value and pass the new value into Excel when it asks for updates. But there are some instances weve heard of where someone wants every update. In that case, the RTD server needs to queue up updates. When Excel asks for updates, the RTD server sends Excel the oldest one in the queue. The server continues calling the **UpdateNotify** method while the queues still have values in them.

Why Aren't My Values Updating?

One of the following is likely the cause:

- The throttle interval is set to -1. This is manual mode, and Excel doesn't check for updates until `Application.RTD.RefreshData` is called.

- The throttle interval is set very high. If the throttle interval is set very high, Excel doesn't check for updates for a long time. By default the throttle interval is set to 2,000 milliseconds.
- A cell is being edited. When Excel is in edit mode, it does not check for updates.
- Excel is busy. If Excel is showing a modal dialog box or is in the middle of a calculation, it does not check for updates.
- The Excel calculation mode is set to manual. In manual mode you don't see updates until a calculation is triggered (similar to pressing F9). To change the Excel calculation mode, on the **Tools** menu, click **Options**, and then click the **Calculation** tab.
- The RTD server is not calling **UpdateNotify**.

Why Do I Always Get #N/A When I Enter My RTD Formulas?

One of the following is likely the cause:

- Macros were disabled. If the RTD server is running locally, under Medium and High security, users see a security warning dialog box. They must enable macros for the RTD server. If the RTD server is not code signed, it just silently fails under High security.
- The RTD server didn't start successfully. The **ServerStart** method must return 1. Otherwise, Excel calls the **ServerTerminate** method because it thinks the RTD server failed to start.
- The RTD server is listed among the disabled items. To check the disabled items, on the **Help** menu, click **About Microsoft Excel**, and then click **Disabled Items**.
- The RTD server didn't return anything in the **ConnectData** method and hasn't called the **UpdateNotify** method yet.

Why Do My RTD Formulas Only Update Once Every Two Seconds?

Excel has the notion of a throttle for RTD. By default this throttle is set at 2,000 milliseconds (two seconds). What this means is that Excel only checks to see if it has been notified of an update at most once every two seconds. If Excel is busy, it may not check it for longer than two seconds, but if is not busy, it basically checks for updates every two seconds. This throttle interval can be modified by adjusting `Application.RTD.ThrottleInterval` in Excel.

How Do I Get My RTD Formulas to Update Faster?

Lower the Excel throttle interval, and/or have the RTD server call the **UpdateNotify** method more frequently.

> **Caution** If updates come in so frequently that Excel is continuously updating values and doing calculations, Excel might end up in a state where it never gives the user a chance to do anything, effectively getting in a hung state. If this happens, set the Excel throttle interval higher.